Django Base Documentation

Release 0.1

Tiago Almeida

Feb 20, 2018

Contents

CHAPTER 1

Django Base

1.1 Documentation

The full documentation is at http://django-base.readthedocs.io.

Live demo @ http://django-base.104.236.104.21.xip.io

1.2 Quickstart

1. Clone o repositório.
2. Crie um virutalenv com o Python 3.5
3. Ative o Virtualenv.
4. Instale as dependencias.

5. Configure as variaveis sensiveis do projeto com o .env 5. Configure as variaveis referentes ao dokku no arquivo deploy_utlis/.env 6. Migre seus modelos para o Banco de Dados 7. Roda o collectstatic para configurar arquivos staticos 8. Execute os testes.

Digite no terminal:

```
git clone https://github.com/tyagow/django-base.git Nome-Do-Projeto
cd Nome-Do-Projeto
python -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
cp contrib/env-sample .env
cp contrib/env-d0kku-sample deploy_utlis/.env
python manage.py collectstatic
python manage.py migrate
```

```
python manage.py test
python manage.py runserver
```

1.3 Como fazer o Deploy?

Como fazer o Deploy?

1.4 Features

- Django 1.10.5
- Bootstrap 4 alpha 6
- JQuery 3.1.1
- Python Decouple
- DJ Static (serving static files locally)
- Dj Database URL
- Django test without migrations
- Django Crispy Forms
- Django bootstrap3
- Social User Login App* (facebook e twitter)
- Django Extensions
- Dokku pre configured
- Multi languange i18n
- Coverage

Need additional configuration

1.5 Social Auth

- **Adicionar ao INSTALLED_APPS**

```
'social_django',
```

- **Adicionar ao settings.py**

```
AUTHENTICATION_BACKENDS = (
  'social_core.backends.twitter.TwitterOAuth',
  'social_core.backends.facebook.FacebookOAuth2',
  'django.contrib.auth.backends.ModelBackend',
)
```

- **Adicionar ao requirements.txt**

```
social-auth-app-django
```

- **Adicionar ao urls.py**

```
url('', include('social_django.urls', namespace='social'))
```

- **Adicionar ao MIDDLEWARE_CLASSES**

```
'social_django.middleware.SocialAuthExceptionMiddleware',
```

- **Adicionar ao TEMPLATES**

```
'social_django.context_processors.backends',
'social_django.context_processors.login_redirect',
```

- **Configurar variaveis no .env e no servidor**

```
SOCIAL_AUTH_TWITTER_KEY=
SOCIAL_AUTH_TWITTER_SECRET=
SOCIAL_AUTH_FACEBOOK_KEY=
SOCIAL_AUTH_FACEBOOK_SECRET=
```

- **Configurar o HOST no App do Facebook**
- **Uncomment buttons to social login in registration/login.html**
- Tutorial: https://simpleisbetterthancomplex.com/tutorial/2016/10/24/how-to-add-social-login-to-django.html

1.6 Translation

- Tutorial: http://www.marinamele.com/taskbuster-django-tutorial/internationalization-localization-languages-time-zones

1.7 Running Tests

Does the code actually work?

```
source .venv/bin/activate
(myenv) $ python manage.py test
```

CHAPTER 2

Installation

1. Clone o repositório.
2. Crie um virutalenv com o Python 3.5
3. Ative o Virtualenv.
4. Instale as dependencias.
5. Configure a instancia com o .env
6. Migre seus modelos para o Banco de Dados
7. Roda o collectstatic para configurar arquivos staticos
8. Execute os testes.

Digite no terminal:

```
git clone https://github.com/tyagow/django-base.git Nome-Do-Projeto
cd Nome-Do-Projeto
python -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
cp contrib/env-sample .env
python manage.py collectstatic
python manage.py migrate
python manage.py test
python manage.py runserver
```

CHAPTER 3

Usage

3.1 Translation

Complilar textos a serem traduzidos

- Criar uma pasta locale dentro de cada App que será traduzido.
- Criar arquivo para as mensagens a serem traduzidas:

```
django-admin makemessages -l en
```

*O ultimo parametro é referente a linguagem que o arquivo será traduzida

- Traduzir o arquivo django.po
- Compliar arquivo traduzido:

```
django-admin compilemessages
```

NOTES

- Não traduzir palavras dentro de {} pois são variáveis usadas pelo django e não texto a ser traduzido.

3.2 Pagination

Pode usar parametros como filtros que os links da paginação irão redirecionar para a pagina certa com o link dos filtros ativos.

Na view:

```
parametros = ''
for item, value in request.GET.dict().items():
    if not item == 'page':
        parametros += '&{}={}'.format(item, value)
```

```
parametros = parametros.replace(' ', '+')
 paginator = Paginator(queryset_list, PAGINACAO_QUANTIDADE_POR_PAGINA)

page_request_var = "page"
page = request.GET.get(page_request_var)
try:
    queryset = paginator.page(page)
except PageNotAnInteger:
    queryset = paginator.page(1)
except EmptyPage:
    queryset = paginator.page(paginator.num_pages)

[...]
context['parametros'] = parametros
 return render(request, [...])
```

No template:

```
{% include 'widgets/pagination.html' with object_list=query_list pagination_section='
↪#section_target' %}
```

Criar o arquivo widgets/pagination.html na pasta templates com o seguinte conteudo:

```
<div class="row">
    {% if object_list.paginator.num_pages > 1 %}
        <ul class="pagination">
            {% with ''|center:object_list.paginator.num_pages as range %}
                {% for _ in range %}
                    <li{% if forloop.counter == object_list.number %} class="active"{
↪% endif %}><a href="?page={{ forloop.counter }}{{ parametros }}{{ pagination_
↪section }}">{{ forloop.counter }}</a></li>
                {% endfor %}
            {% endwith %}
        </ul>
    {% endif %}
</div>
```

CHAPTER 4

Como fazer o Deploy?

1. Install Digital Ocean Dokku image
2. Send your ssh-key to dokku
3. Connect via ssh to your server
4. Create app in dokku
5. Install postgres plugin in dokku
6. Create database for your app in dokku
7. Link database and app in dokku
8. Set DEBUG in dokku
9. Generate new SECRET_KEY
10. Set SECRET_KEY in dokku
11. Set ALLOWED_HOSTS in dokku
12. Set Global Domain dor dokku
13. Push your code to dokku
14. Run the migrations
15. Collect static data with DEBUG=False

Digite no terminal

```
(local) cat ~/.ssh/id_rsa.pub | ssh root@<your.ip.address> "sudo sshcommand acl-add␣
↪dokku [description]"
(local) ssh root@<your.ip.address>
(server) dokku apps:create <app-name>
(server) sudo dokku plugin:install https://github.com/dokku/dokku-postgres.git
(server) dokku postgres:create <database-name>
(server) dokku postgres:link <databse-name> <app-name>
(local) git remote add dokku dokku@dokku.me:<app-name>
```

```
(local) ssh dokku@<your.ip.address> config:set <app-name> DEBUG=False
(local) python contrib/secret_gen.py
(local) ssh dokku@<your.ip.address> config:set <app-name> SECRET_KEY='<new-generated-
↪key>'
(local) ssh dokku@<your.ip.address> config:set <app-name> ALLOWED_HOSTS=<app-name>.
↪<your.ip.address>.xip.io
(local) ssh dokku@<your.ip.address> config:set <app-name> AWS_STORAGE_BUCKET_
↪NAME=XXXXXXXXXXX AWS_ACCESS_KEY_ID=XXXXXXXXXXX AWS_SECRET_ACCESS_KEY=XXXXXXXXXXX
(local) ssh dokku@<your.ip.address> domains:add-global <your.ip.address>.xip.io
(local) ssh dokku@<your.ip.address> domains:enable <app-name>
(local) git push dokku master
(local) ssh dokku@<your.ip.address> run <app-name> python manage.py migrate
(local) DEBUG=False python manage.py collectstatic
```

Note:

- Depois do primeiro deploy feito basta um comando para o deploy:

```
git push dokku master
```

- Não esquecer de migrar/atualizar o banco de dados sempre que alterar um modelo:

```
ssh dokku@<your.ip.address> run <app-name> python manage.py migrate
```

- http://dokku.viewdocs.io/dokku/deployment/application-deployment/

CHAPTER 5

Dokku

- Change PORT

‘ (não recomendado, se configurar na porta 80 só poderei ter 1 serviço (app)) ” you can only bind a single service to port 80 if you do not use a vhost but i highly suggest using a vhost for your server so then you get urls like app.vhost.com ” @ savant‘

‘ dokku config:set APP DOKKU_NGINX_PORT=80 DOKKU_PROXY_PORT_MAP=http:80:5000 ‘

- **Configurar um vhost**

`dokku domains:add-global domain_here`

- **Re-enable vhosts for your app**

(http://dokku.viewdocs.io/dokku/configuration/domains/) `dokku domains:enable APP`

- **Server < 1 GB RAM**
- http://dokku.viewdocs.io/dokku/getting-started/advanced-installation/#vms-with-less-than-1gb-of-memory

Run on server:

```
cd /var
touch swap.img
chmod 600 swap.img

dd if=/dev/zero of=/var/swap.img bs=1024k count=1000
mkswap /var/swap.img
swapon /var/swap.img
free

echo "/var/swap.img    none    swap    sw    0    0" >> /etc/fstab
```

CHAPTER 6

Configurar AmazonS3

- https://www.caktusgroup.com/blog/2014/11/10/Using-Amazon-S3-to-store-your-Django-sites-static-and-media-files/

CHAPTER 7

Circle-ci Automatic Deploy to Dokku

1. Generate new ssh-key without password - Give a name to the file.
2. Copy new ssh-key.pub to your project (you must be in your root project)
3. Add ssh-key private to circle-ci in circle-ci website
4. Edit circle.yml
5. add ssh-key to dokku server (go to add-ssh-to-dokku section)

Terminal:

```
$ ssh-keygen -t rsa
$ cp ~/.ssh/<ssh-key>.pub ./deploy_utils/deploy_key
$ circle.yml ->
        machine:
          python:
            version: 3.5.1
        general:
          artifacts:
            - "htmlcov"
        dependencies:
          pre:
          - cp contrib/env-sample .env
        test:
          override:
            - coverage run manage.py test
          post:
            - coverage html -d $CIRCLE_ARTIFACTS
        deployment:
          production:
            branch: master
            commands:
              - git remote add deploy dokku@<server ip>:<dokku app name>
              - git push deploy maste
```

CHAPTER 8

Travis-ci Automatic Deploy to Dokku

1. Generate new ssh-key without password - Give a name to the file.
2. Copy new ssh-key.pub to your project (you must be in your root project)
3. Install Travis CI Command Line Client
4. Login to travis
5. Encrypt new ssh key (private) with travis and copy the command to decrypt in terminal
6. Edit .travis.yml
7. add ssh-key to dokku server (go to add-ssh section)

http://tannguyen.org/2017/02/set-up-hugo-dokku-and-travis/

Terminal:

```
$ ssh-keygen -t rsa
$ cp ~/.ssh/<ssh-key>.pub ./deploy_utils/deploy_key
$ sudo gem install travis
$ travis login
$ travis encrypt-file deploy_utils/deploy_key
->[.travis.yml] add before install:
    before_install:
      - openssl aes-256-cbc -K $encrypted_5d3fad67a2c7_key -iv $encrypted_
↪5d3fad67a2c7_iv -in deploy_utils/deploy_key.enc -out deploy_utils/deploy_key -d
      (this command should be given after run travis encrypt-file [...]
->[.travis.yml] add after script:
    after_success:
      - eval "$(ssh-agent -s)" #start the ssh agent
      - chmod 600 deploy_utils/deploy_key # this key should have push access
      - ssh-add deploy_utils/deploy_key
      - echo -e "Host <hostname here>\n\tStrictHostKeyChecking no\n" >> ~/.ssh/config
      - git remote add deploy <git-remote>
      - git push deploy
```

8.1 Errors:

1. installing travis via *sudo gem install travis*

   ERROR: Error installing travis: ERROR: Failed to build gem native extension.

Solution - https://github.com/travis-ci/travis.rb/issues/391

```
sudo apt-get install python-software-properties
sudo apt-add-repository ppa:brightbox/ruby-ng
sudo apt-get update
sudo apt-get install ruby2.1 ruby-switch
sudo ruby-switch --set ruby2.1
sudo apt-get install ruby2.1-dev
```

CHAPTER 9

Add ssh-key to dokku

1. Create a ssh-key
2. send ssh-key to server

Terminal:

```
ssh-keygen -t rsa
cat <path to ssh-key> | ssh root@<your.ip.address> "sudo sshcommand acl-add dokku␣
↪[description]"
```

CHAPTER 10

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

10.1 Types of Contributions

10.1.1 Report Bugs

Report bugs at https://github.com/tyagow/django-base/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

10.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

10.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

10.1.4 Write Documentation

django-base could always use more documentation, whether as part of the official django-base docs, in docstrings, or even on the web in blog posts, articles, and such.

10.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/tyagow/django-base/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

10.2 Get Started!

Ready to contribute? Here's how to set up *django-base* for local development.

1. Clone o repositório.
2. Crie um virutalenv com o Python 3.5
3. Ative o Virtualenv.
4. Instale as dependencias.
5. Configure a instancia com o .env
6. Migre seus modelos para o Banco de Dados
7. Roda o collectstatic para configurar arquivos staticos
8. Execute os testes.

Digite no terminal:

```
git clone https://github.com/tyagow/django-base.git Nome-Do-Projeto
cd Nome-Do-Projeto
python -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
cp contrib/env-sample .env
python manage.py collectstatic
python manage.py migrate
python manage.py test
python manage.py runserver
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

6. Submit a pull request through the GitHub website.

10.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, and 3.5, and for PyPy. Check https://travis-ci.org/tyagow/django-base/pull_requests and make sure that the tests pass for all supported Python versions.

10.4 Tips

To run a subset of tests:

```
$ python manage.py test
```

CHAPTER 11

Credits

11.1 Development Lead

- django-base <tyagow@hotmail.com.br>

11.2 Contributors

None yet. Why not be the first?

CHAPTER 12

History

12.1 0.1.0 (11-2-2017)

- Projeto django base para desenvolvimento web, com features básicas.
- Suporte ao Amazons3
- Inicio do historico

CHAPTER 13

Indices and tables

- genindex
- modindex
- search